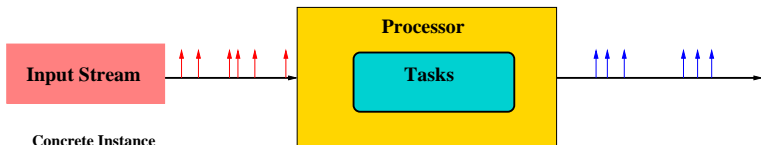

Real-Time Calculus

Prof. Dr. Jian-Jia Chen

LS 12, TU Dortmund

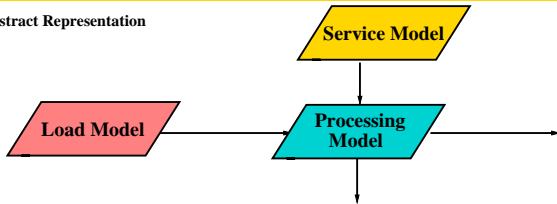
10, Dec., 2019

Abstract Models for Real-Time Calculus

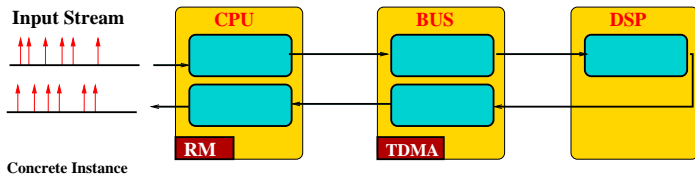


Concrete Instance

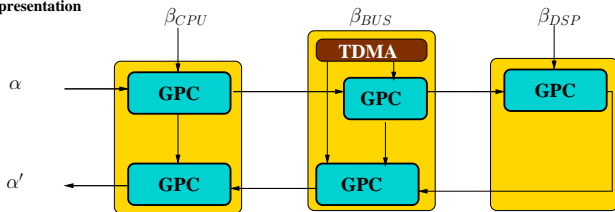
Abstract Representation



Abstract Models for Module Performance Analysis



Abstract Representation



RM: Rate-Monotonic (a fixed-priority scheduler, detailed in Chapter 6)

TDMA: Time Division Multiple Access (detailed later)

GPC: Greedy Processing Component (detailed later)

Overview

System View

Module Performance Analysis (MPA)

Math. View

Real-Time Calculus (RTC)

Min-Plus Calculus, Max-Plus Calculus

Backgrounds

- Real-Time Calculus can be regarded as a worst-case/best-case variant of classical queuing theory. It is a formal method for the analysis of distributed real-time embedded systems.
- Related Work:
 - Min-Plus Algebra: F. Baccelli, G. Cohen, G. J. Olster, and J. P. Quadrat, Synchronization and Linearity —An Algebra for Discrete Event Systems, Wiley, New York, 1992.
 - Network Calculus: J.-Y. Le Boudec and P. Thiran, Network Calculus -A Theory of Deterministic Queuing Systems for the Internet, Lecture Notes in Computer Science, vol. 2050, Springer Verlag, 2001.

Definition of Arrival Curves and Service Curves

- For a specific trace:
 - Data streams: $R(t)$ = number of events in $[0, t)$
 - Resource stream: $C(t)$ = available resource in $[0, t)$
- For the worst cases and the best cases in any interval with length Δ :
 - Arrival Curve $[\alpha^l, \alpha^u]$:

$$\alpha^l(\Delta) = \inf_{\lambda \geq 0, \forall R} \{R(\Delta + \lambda) - R(\lambda)\}$$

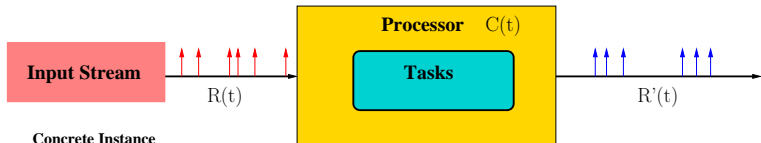
$$\alpha^u(\Delta) = \sup_{\lambda \geq 0, \forall R} \{R(\Delta + \lambda) - R(\lambda)\}$$

- Service Curve $[\beta^l, \beta^u]$:

$$\beta^l(\Delta) = \inf_{\lambda \geq 0, \forall C} \{C(\Delta + \lambda) - C(\lambda)\}$$

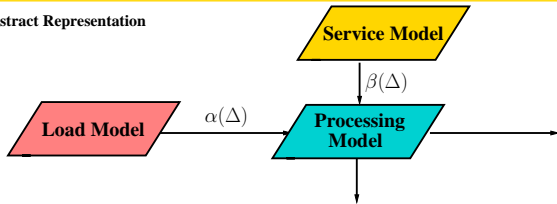
$$\beta^u(\Delta) = \sup_{\lambda \geq 0, \forall C} \{C(\Delta + \lambda) - C(\lambda)\}$$

Abstract Models for Real-Time Calculus



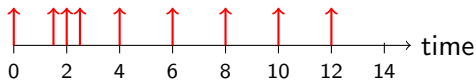
Concrete Instance

Abstract Representation



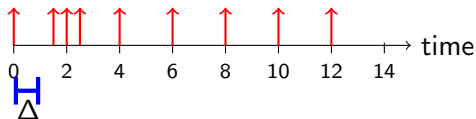
Arrival Curve: An Example

Use a sliding window to get the upper bound of the number of events in a specified interval length.



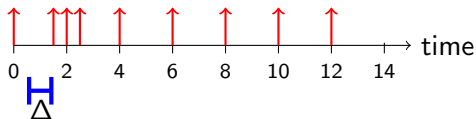
Arrival Curve: An Example

Use a sliding window to get the upper bound of the number of events in a specified interval length.



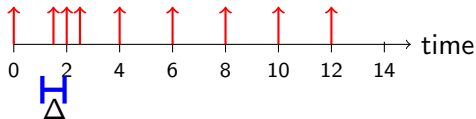
Arrival Curve: An Example

Use a sliding window to get the upper bound of the number of events in a specified interval length.



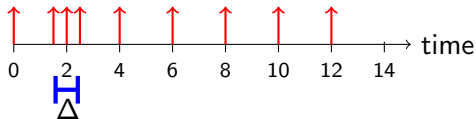
Arrival Curve: An Example

Use a sliding window to get the upper bound of the number of events in a specified interval length.



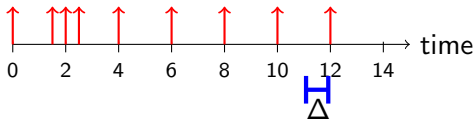
Arrival Curve: An Example

Use a sliding window to get the upper bound of the number of events in a specified interval length.



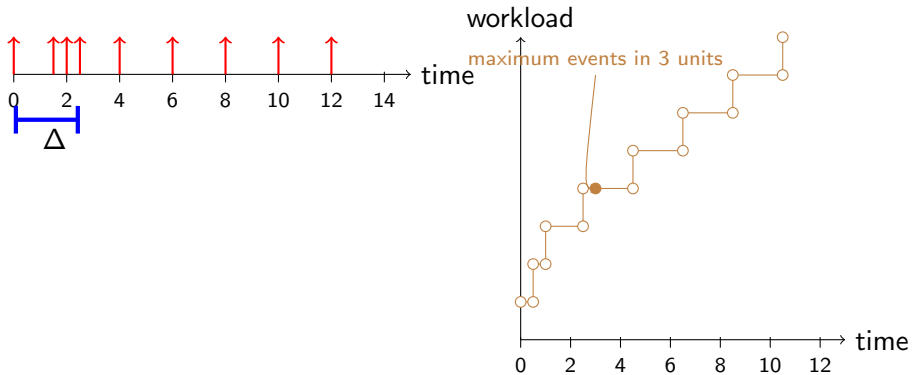
Arrival Curve: An Example

Use a sliding window to get the upper bound of the number of events in a specified interval length.



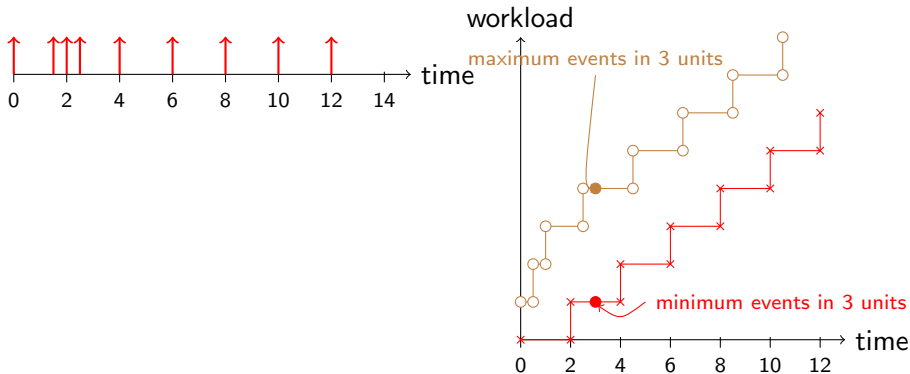
Arrival Curve: An Example

Use a sliding window to get the upper bound of the number of events in a specified interval length.



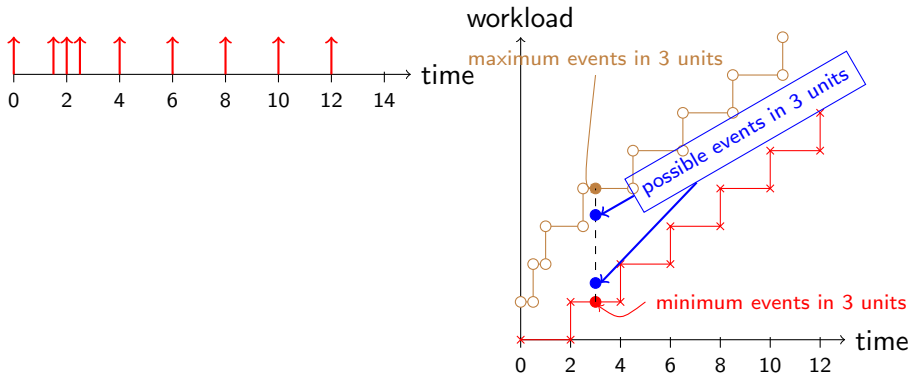
Arrival Curve: An Example

Use a sliding window to get the upper bound of the number of events in a specified interval length.



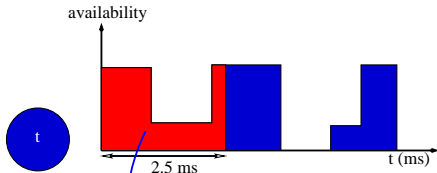
Arrival Curve: An Example

Use a sliding window to get the upper bound of the number of events in a specified interval length.

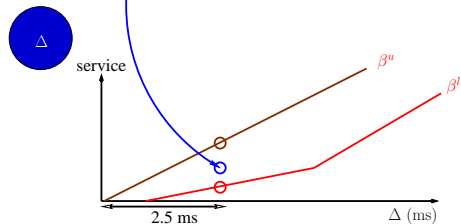


Service Curve: An Example

Resource
Availability

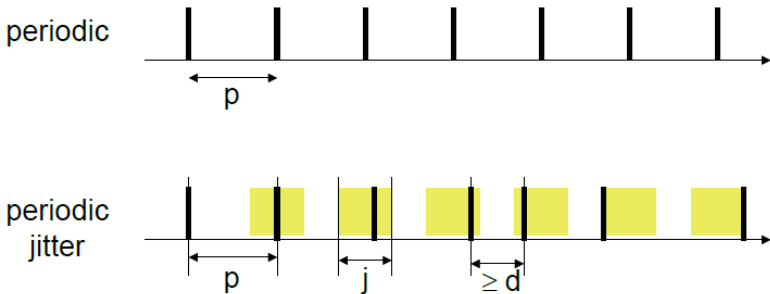


Service Curves
 $\beta = [\beta^l, \beta^u]$



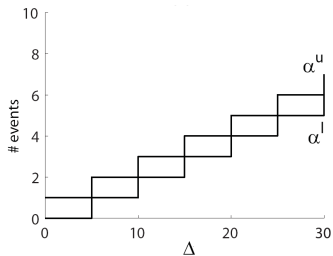
Example 1: Periodic with Jitter

A common event pattern that is used in literature can be specified by the parameter triple (p, j, d) , where p denotes the period, j the jitter, and d the minimum inter-arrival distance of events in the modeled stream.



Example 1: Periodic with Jitter

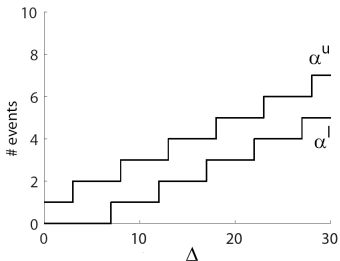
Periodic



$$\alpha^u(\Delta) = \left\lceil \frac{\Delta}{p} \right\rceil$$

$$\alpha^l(\Delta) = \left\lfloor \frac{\Delta}{p} \right\rfloor$$

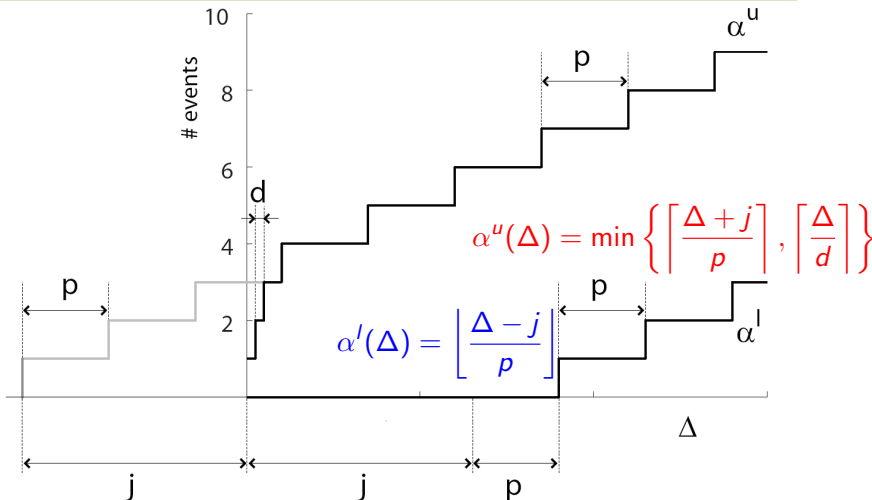
Periodic with Jitter



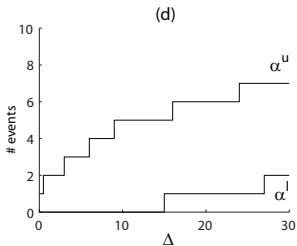
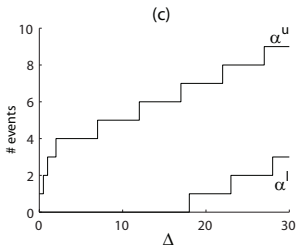
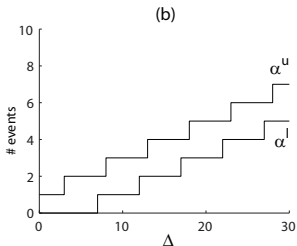
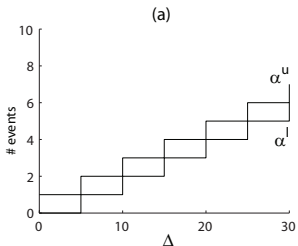
$$\alpha^u(\Delta) = \left\lceil \frac{\Delta + j}{p} \right\rceil$$

$$\alpha^l(\Delta) = \left\lfloor \frac{\Delta - j}{p} \right\rfloor$$

Example 1: Periodic with Jitter

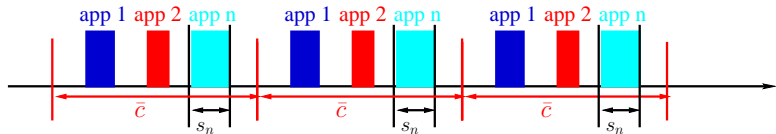


More Examples on Arrival Curves

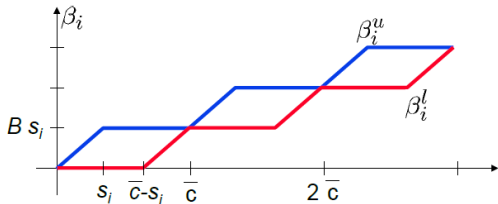
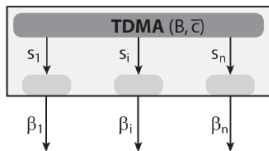


Example 2: TDMA Resource

- Consider a real-time system consisting of n applications that are executed on a resource with bandwidth B that controls resource access using a TDMA (Time Division Multiple Access) policy.
- Analogously, we could consider a distributed system with n communicating nodes, that communicate via a shared bus with bandwidth B , with a bus arbitrator that implements a TDMA policy.
- TDMA policy: In every TDMA cycle of length \bar{c} , one single resource slot of length s_i is assigned to application i .



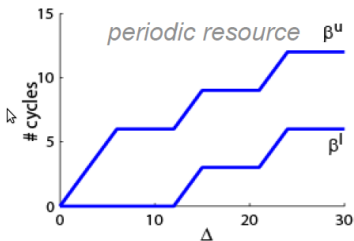
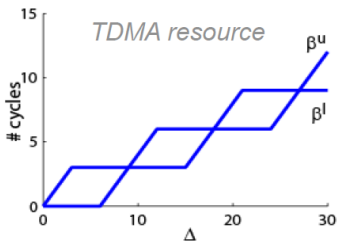
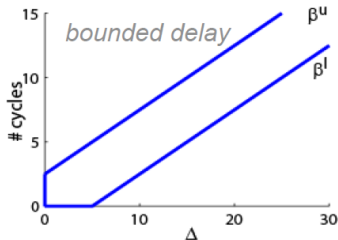
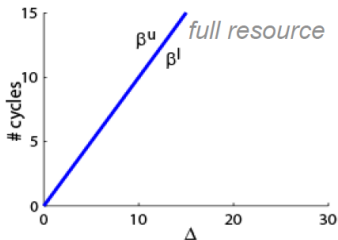
Example 2: TDMA Resource



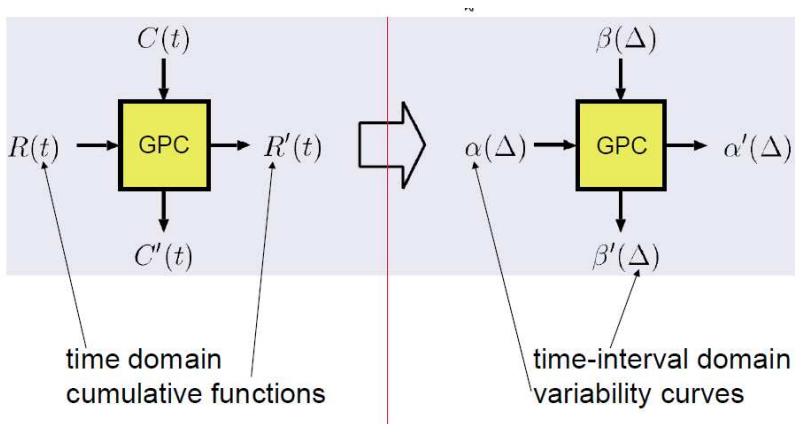
$$\beta^u(\Delta) = B \min \left\{ \left\lfloor \frac{\Delta}{\bar{c}} \right\rfloor s_i, \Delta - \left\lfloor \frac{\Delta}{\bar{c}} \right\rfloor (\bar{c} - s_i) \right\}$$

$$\beta^l(\Delta) = B \max \left\{ \left\lfloor \frac{\Delta}{\bar{c}} \right\rfloor s_i, \Delta - \left\lfloor \frac{\Delta}{\bar{c}} \right\rfloor (\bar{c} - s_i) \right\}$$

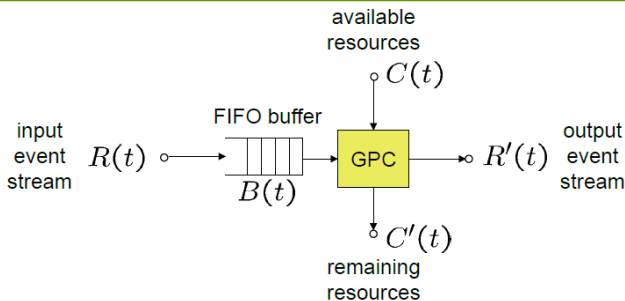
More Examples on Service Curves



Abstraction



Greedy Processing Component (GPC)



- Component is triggered by incoming events.
- A fully preemptable task is instantiated at every event arrival to process the incoming event.
- Active tasks are processed in a greedy fashion in FIFO order.
- Processing is restricted by the availability of resources.

Some Relations (only for your reference)

- The output upper arrival curve of a component satisfies

$$\alpha^{u'} \leq (\alpha^u \circ \beta^l)$$

with a simple and pessimistic calculation.

- The remaining lower service curve of a component satisfies

$$\beta^{l'}(\Delta) = \sup_{0 \leq \lambda \leq \Delta} (\beta^l(\lambda) - \alpha^u(\lambda))$$

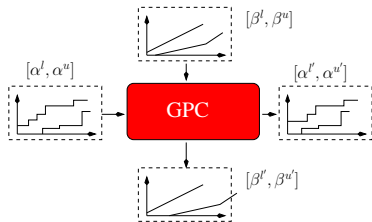
More Relations (only for your reference)

$$\alpha^{u'} = [(\alpha^u \otimes \beta^u) \otimes \beta^l] \wedge \beta^u$$

$$\alpha^{l'} = [(\alpha^u \otimes \beta^l) \otimes \beta^l] \wedge \beta^l$$

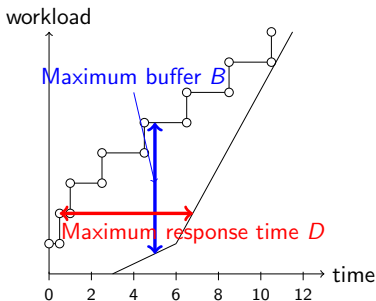
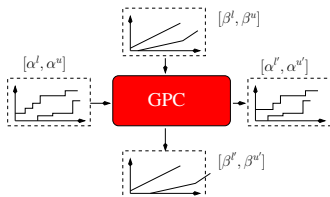
$$\beta^{u'} = (\beta^u - \alpha^{l'}) \bar{\otimes} 0$$

$$\beta^{l'} = (\beta^l - \alpha^u) \bar{\otimes} 0$$



Without formal proofs....

Graphical Interpretation

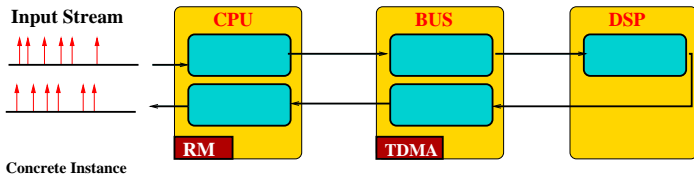


$$B = \sup_{t \geq 0} \{R(t) - R'(t)\} \leq \sup_{\lambda \geq 0} \{\alpha^u(\lambda) - \beta^l(\lambda)\}$$

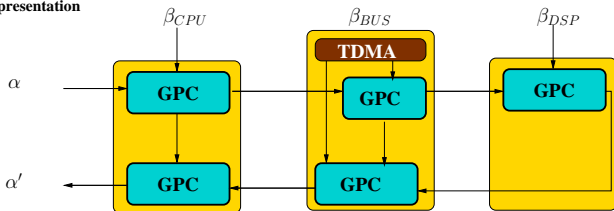
$$D = \sup_{t \geq 0} \{\inf\{\tau \geq 0 : R(t) \leq R'(t + \tau)\}\}$$

$$= \sup_{\Delta \geq 0} \{\inf\{\tau \geq 0 : \alpha^u(\Delta) \leq \beta^l(\Delta + \tau)\}\}$$

Complete System Composition



Abstract Representation



RM: Rate-Monotonic (a fixed-priority scheduler, detailed in Chapter 6)

TDMA: Time Division Multiple Access

GPC: Greedy Processing Component

RTC Toolbox (<http://www.mpa.ethz.ch/Rtctoolbox>)

Modular Performance Analysis with Real-Time Calculus

Rtctoolbox :: Overview

View Edit History Print

Overview

- RTC Toolbox**
 - Overview
 - Download
 - Release Notes
 - User Guide
 - FAQ
- RTS Toolbox**
 - Overview
 - Download
 - Release Notes
- PESIMDES**
 - Overview
 - Modeling Scope
 - Download & Setup
 - User Guide
- Publications**
 - Overview
 - Student Theses
- Wiki**
 - Search
 - WikiSandbox

edit SideBar

Real-Time Calculus Toolbox

Overview

The Real-Time Calculus (RTC) Toolbox is a free Matlab toolbox for system-level performance analysis of distributed real-time and embedded systems.

The RTC Toolbox is based on an efficient representation of Variability Characterization Curves (VCC's) and implements most min-plus and max-plus algebra operators for these curves. On top of the min-plus and max-plus algebra operators, the RTC Toolbox provides a library of functions for Modular Performance Analysis with Real-Time Calculus.

Latest News

- [2010-07-26]: [Interface to SymTA/S analysis tool.](#)
- [2010-07-26]: [Extensions for structured event streams.](#)
- [2009-01-30]: [BugFix and Update released.](#)
- [2008-12-23]: [Beta Version 1.2 released.](#)
- [2008-10-14]: [BugFix released.](#)
- [2008-07-16]: [BugFix released.](#)
- [2008-05-30]: [BugFix released.](#)
- [2008-02-06]: [BugFix released.](#)
- [2007-09-24]: [New components and tutorial.](#)
- [2007-07-05]: [BugFix released.](#)
- [2007-06-25]: [BugFix released.](#)
- [2007-06-21]: [New Version released.](#)
- [2007-03-21]: [BugFix released.](#)
- [2006-10-02]: [New tutorials and Java API released.](#)
- [2006-10-02]: [BugFix released.](#)
- [2006-04-04]: [First tutorial published.](#)
- [2006-02-27]: [First official beta version released.](#)

Advantages and Disadvantages of RTC and MPA

- Advantages
 - Provides a powerful abstraction to model event arrivals and resource consumption
 - Considers resources as first-class citizens
 - Allows composition in terms of (a) tasks, (b) streams, (c) resources, (d) sharing strategies.
- Disadvantages
 - Needs some effort to understand and implement
 - Extension to new arbitration schemes not always simple
 - *Not applicable for schedulers that change the scheduling policies dynamically.*